University of Virginia
Charles L. Brown Department of Electrical and Computer Engineering

Project SRAM - Design Review 2

Report to PICo Review Board

By: Austin Moran, Xiafei Yang, and Mark Cheung

SUBJECT: Progress on designing a 1Mb low-power SRAM

Review:

This report shows the progress that we have made following the proposal to design a 1Mb low-power SRAM to meet PICo's specifications. Our primary goal is to minimize the total power with reasonable sizing and delay. Our secondary goal is to consider as many special features as possible by looking at the array of research in this area. A typical structure for an SRAM contains decoders, memory array, sense amplifier and periphery circuit to access the bit cells. Energy will be consumed by all these components. The key metric to optimize energy consumption is (Active Energy per Access)^2*Delay*Area*Idle Power. In order to win the contract of Portable Instruments Company (PICo), we are mainly focused on low-power techniques.

At this point in time we have made quite a bit of progress on the project. Further progress will be dependent on consulting PICo to resolve issues with simulation and modeling the components necessary for bitcell functionality simulation (dummy cells and bitline capacitance, etc.)

In this paper we have revised our old timeline and offer a new, more detailed one for next period. We also have an updated block diagram of the SRAM and a lists of status of each component of the SRAM.

Scheduling:

Below are the timelines we initially planned and lately updated for challenges and setbacks. Figure 1 shows the old schedule with annotations about discrepancies with real circumstances. Figure 2 is the updated timeline with adjustments for the incomplete tasks of this project. Figure 3 presents the separate responsibilities for each team member.

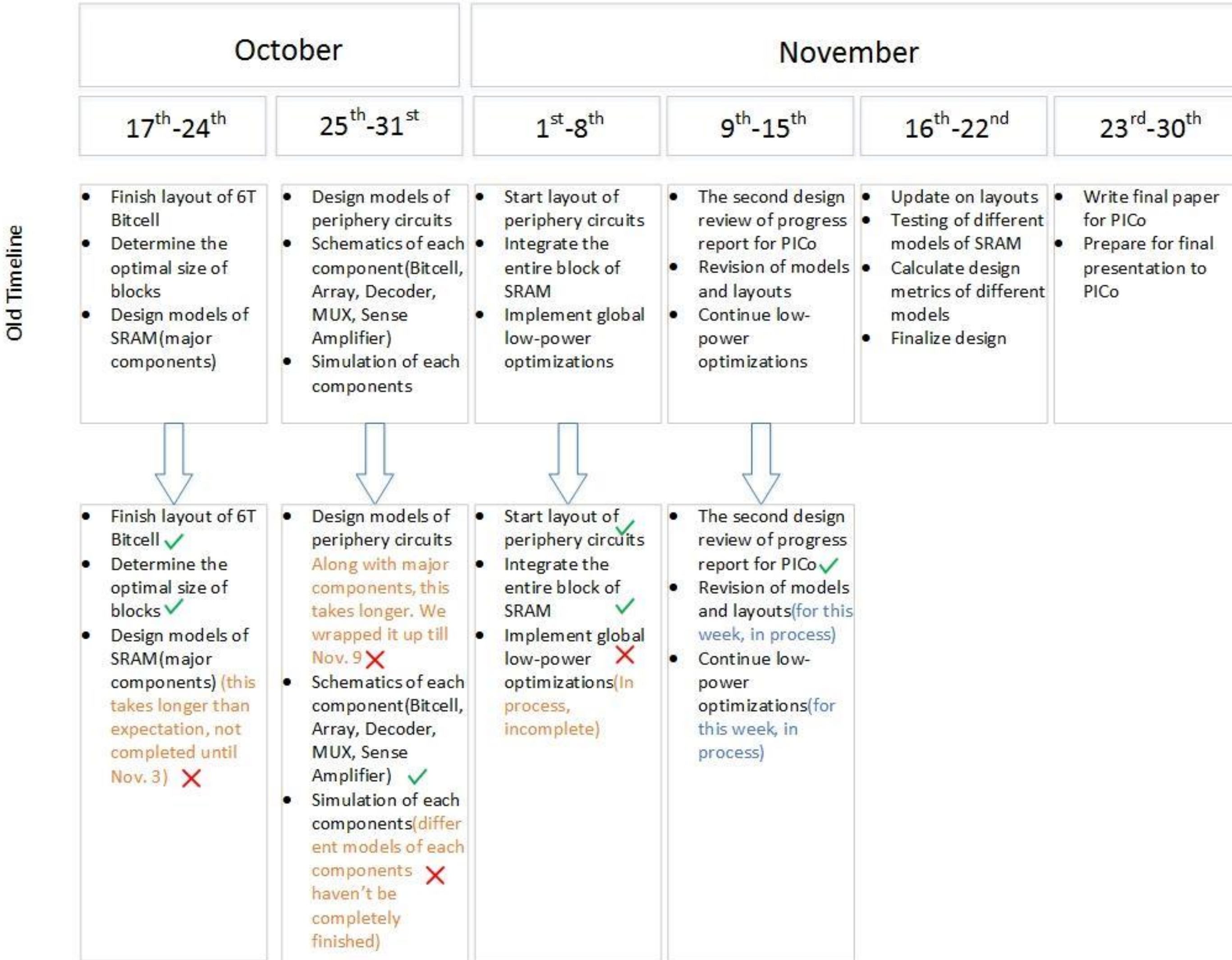


Figure 1. Old timeline with comments on progress compared to the original plan

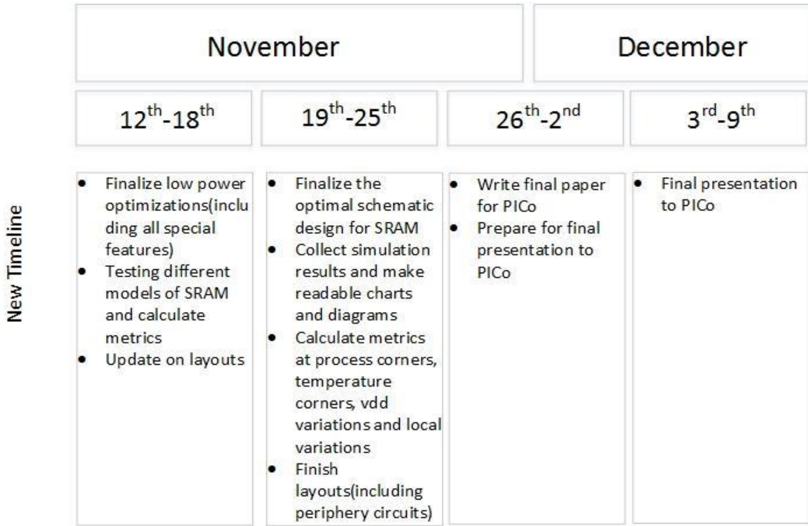


Figure 2. Updated timeline with remaining tasks to complete till the final presentation

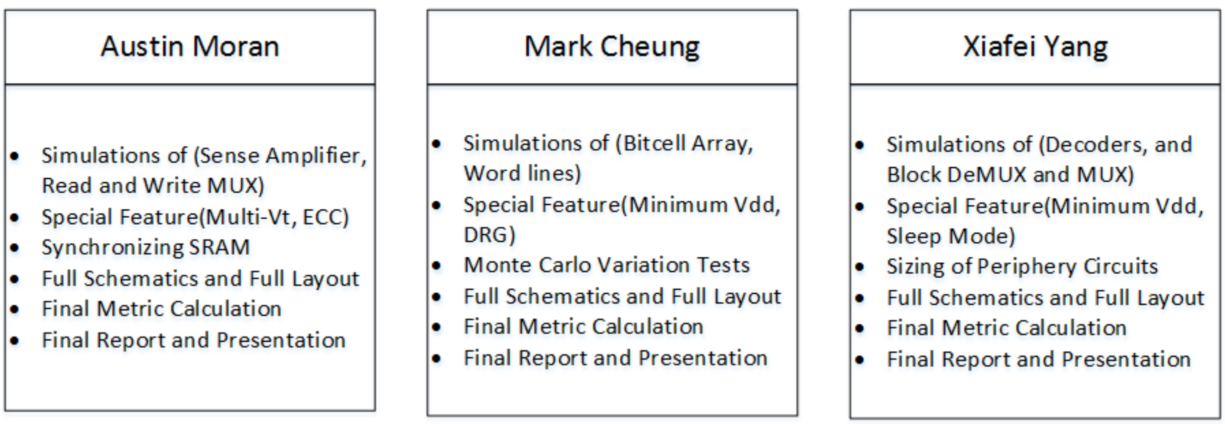


Figure 3.Updated breakdown tasks for each group member

Block Diagram

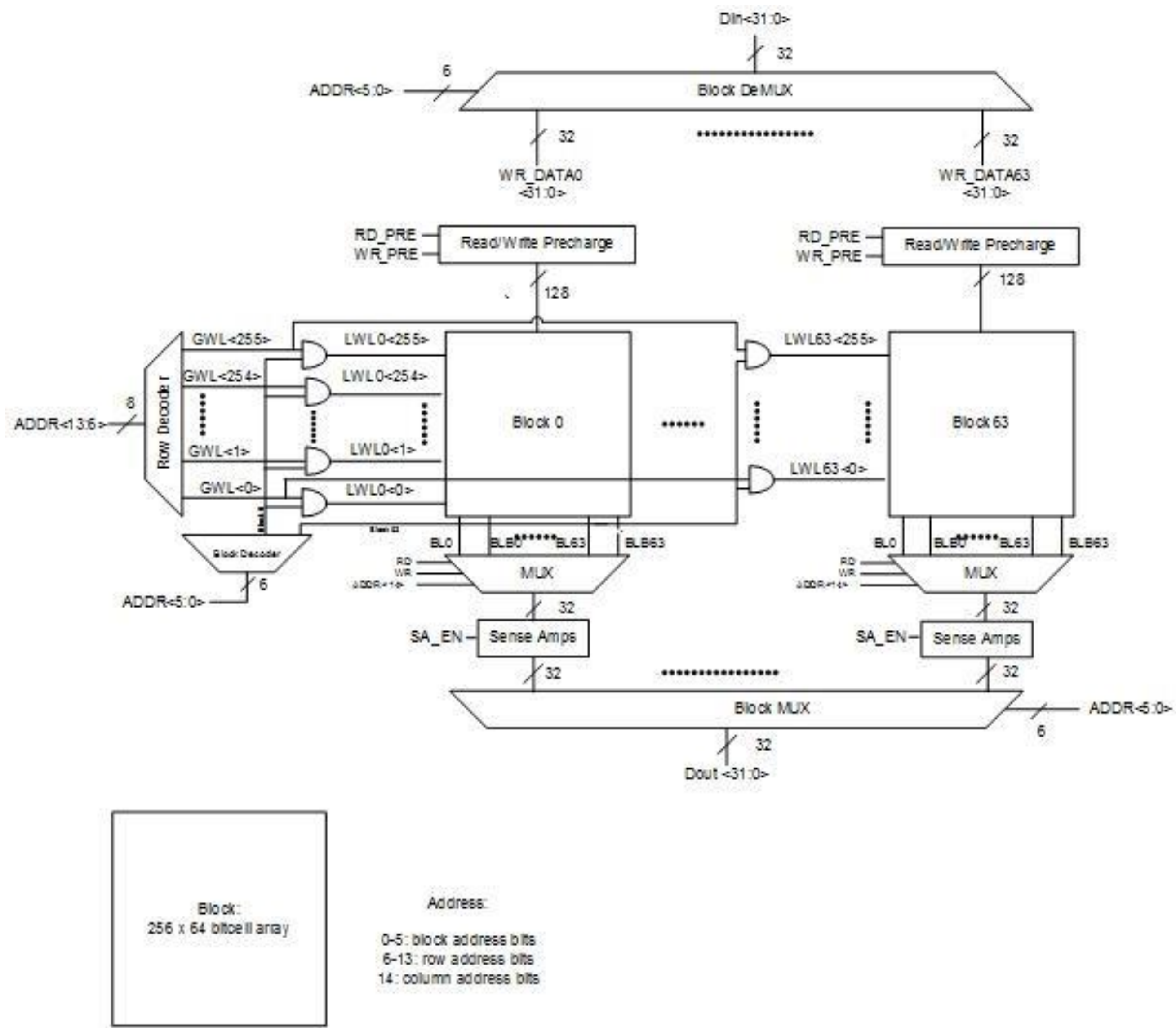


Figure 4.Block diagram of the entire SRAM

Component status list:

Component	Status of Schematic	Status of Layout	Status of Model	Status of Simulation
Precharge	Incomplete	Incomplete	Complete	Complete
Block	Complete	Complete	Complete	Complete
Block Decoder	Complete	Incomplete	Complete	Complete

Row Decoder	Complete	Incomplete	Complete	Complete
Column Decoder	Complete	Incomplete	Complete	Complete
Word Lines	Incomplete	Incomplete	Complete	Complete
Read/Write Mux	Incomplete	Incomplete	Complete	Complete
Sense Amps	Complete	Incomplete	Complete	Complete
Block DeMUX	Incomplete	Incomplete	Complete	Complete
Block MUX	Incomplete	Incomplete	Complete	Complete

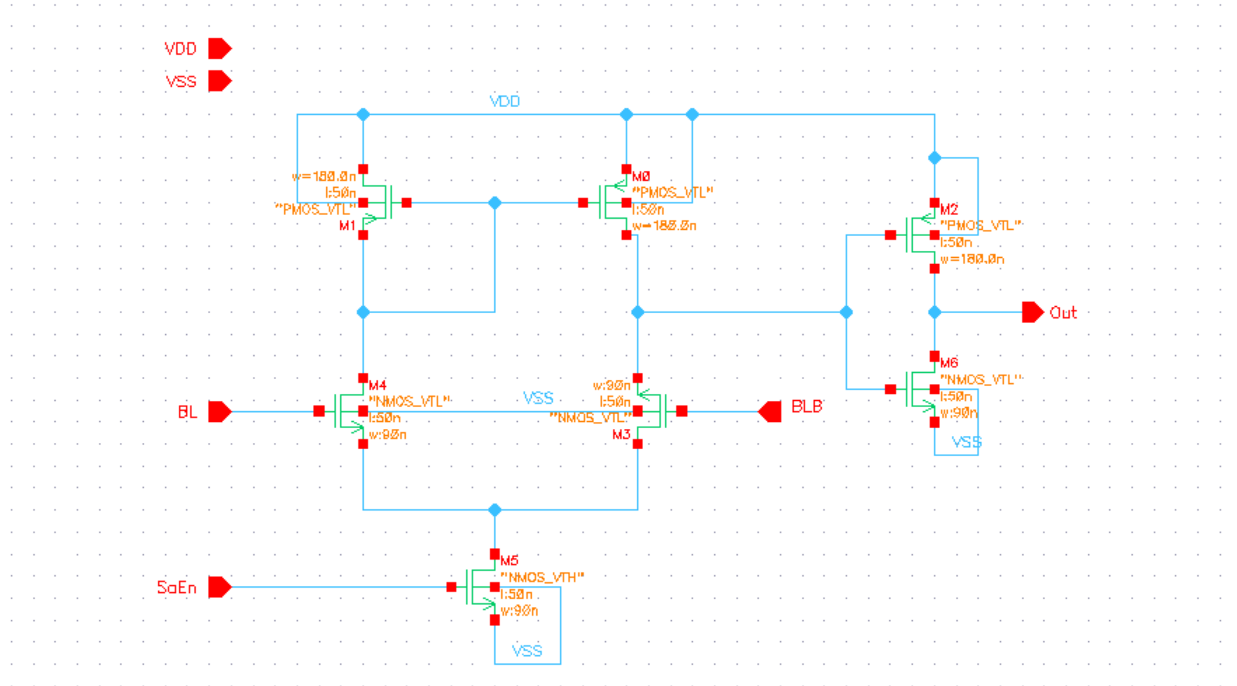
Simulation model

Sense amplifiers

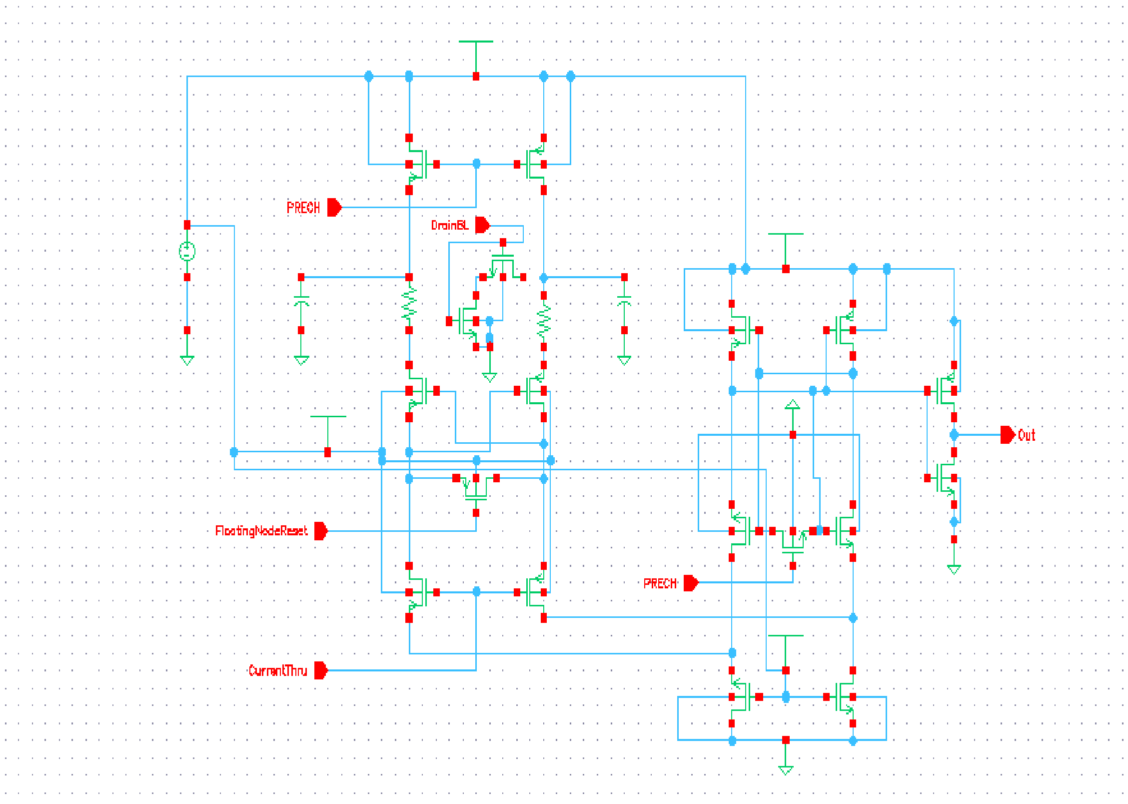
The schematic for the hybrid current sense amplifier using a clamped bit line is complete. However, there are some issues that need to be discussed with the professor. Mainly that the bit lines are not the same voltage after precharging but after being discharged they both return to the same voltage (albeit a lower voltage than they initialize at). A model simulating a loaded bitline is used to simulate/test the sense amplifiers. It consists of precharging transistors that feed into a small network of capacitors and resistors designed to simulate bit lines loaded by bitcells. On one of the bitlines there are two NMOS transistors to simulate the pull down of the bitline that results from reading a zero. The current from the bitlines is then fed into the current sense amplifier. In place of the slightly glitchy hybrid current sense amplifier I have provided a simple voltage sense amplifier. This is to be used to test the functionality of a small (2x2) array of bitcells required for this design review. As well it will provide a nice comparison to and illustrate the savings of implementing a current sense amplifier with a clamped bitline (our proposed sense amplifier). We are choosing to implement the clamped bit line current sense amplifier with a current conveyor because it is capable of operating at a lower supply voltage and the clamped bitline (two NMOS biased in linear region) eliminates the effects of bitline capacitance on sensing speed.[5][6] This results in less delay and, as a result, less energy consumed by the discharging of a bitline.

The remaining technical challenges for the sense amplifiers will be to understand the source of variation in bitline voltage after precharging for the second time. The problem does appear to resolve after the first cycle, so the solution might be to simply perform a dummy read operation before taking data or using the sense amplifier to actually read. In addition, there is some issue with the output of the sense amplifier changing a bit before it is enabled. To resolve this a gating transistor should be placed between VDD and the transistors previously connected to

VDD. This would make the sense amplifier fully power gated instead of just between the sense amplifier and ground and should have the effect of preventing any output swing before activation.



Current sense amplifier(using in the simulation)



Proposed sense amplifier schematics (please see proposal for more information)
This also includes the simulated bitlines and NMOSs to pull down a bitline.

Decoders:

Following the decision on a block size(number of rows and columns) and number of blocks (see block diagram), we added the decoders to our model. There are 64 columns and 256 rows per block in our SRAM, so the column decoder contains one bit, simply selects between the two words. To reduce the capacitance of the address lines to a row decoder and the word-line RC delay, a two-stage hierarchical row decoder is adopted. Figure 5 presents the models for two-level decoders. So we are going to use two 4:16 predecoders as the first level each decoding 4 address bits to drive one of the 16 predecode lines. And the next level ANDs two of the predecode lines to generate the wordline. Row decoder is controlled by the 8-bit address to drive 256 word lines. An AND gate is used with the word line outputs of the decoder and a block select bit from a 6:64 decoder to determine which of the 64 blocks to access at a time, which also represents the predecoding and final decoding stages.[7]

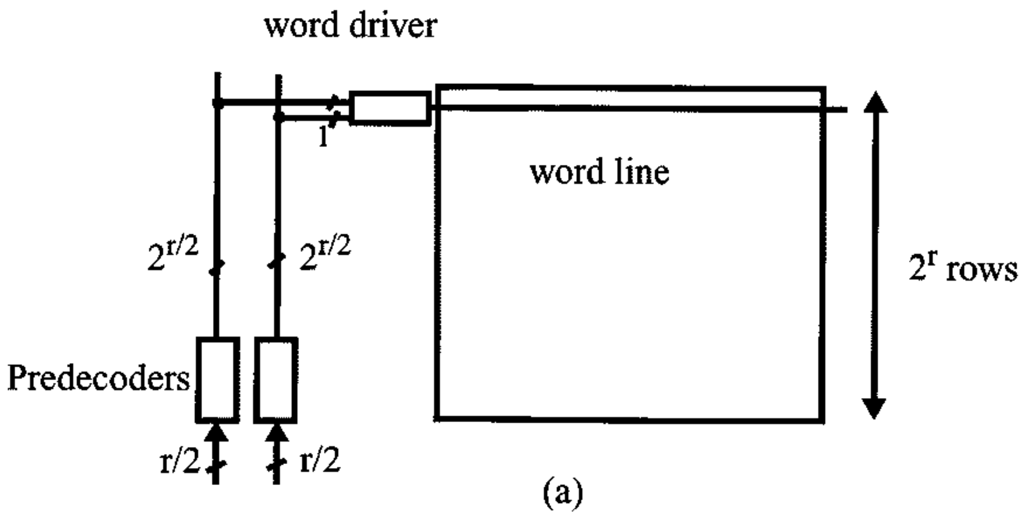


Figure 5.Block diagram for two-level decoders

Multi-Vt: A significant cause of leakage power is due to subthreshold leakage current, which has a direct dependence on Vt: Ileakage= Io (W/L)10^(Vgs-Vt)/S. As Vt decreases, subthreshold leakage increases exponentially. [1] suggests using multiple-threshold CMOS circuits, specifically transistor with higher threshold voltage (NMOS_VTH and PMOS_VTH models). The trade-offs include reduction in gate's drive strengths that results in increased delay. Consequently, we intend to apply this method only on non-critical paths so as to not increase the critical path delay. See below table and simulation. We will need to recalculate the SNM as what we have done are for low threshold transistors only (couldn't find scs models for high threshold transistors).

Transistor	VTH0 (from model specification i.e. *.inc files)
NMOS_VTL	0.3220 V
NMOS_VTG	0.4106 V
NMOS_VTH	0.6078 V

Table 1 (proposal)

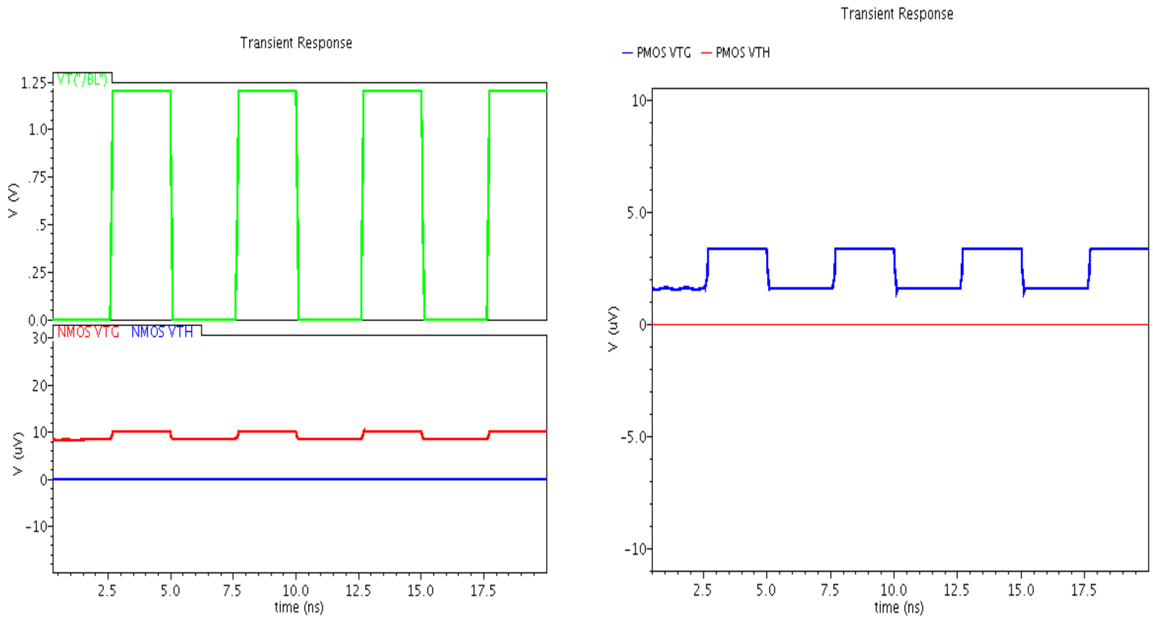
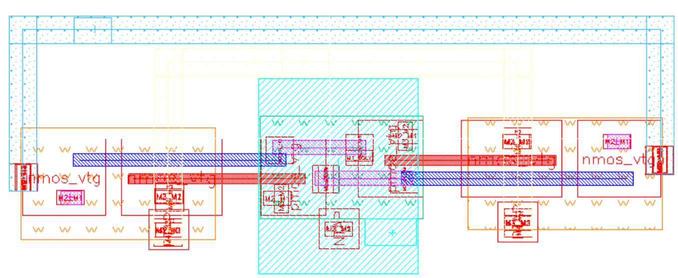


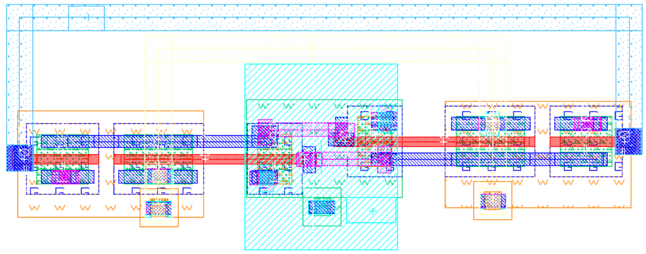
Figure 3a (top left) The voltage of the first graph shows the bitline (due to precharging). The wordline and the data line are both 0, so the bit line and the bit inverse line switches according to the precharge. However, there is leakage across the pass-transistors even during cutoff. The graphs below shows that as the threshold voltage increases (VTG to VTH), the current decreases (lower voltage in the 2nd graph across the inverting NMOS). The effect due to DIBL is shown to be insignificant (when bitline is high). Figure 3b (top right) shows similar happenings for varying the threshold voltage for PMOS. (proposal)

Layout: Single Bitcell & 4 Bitcells

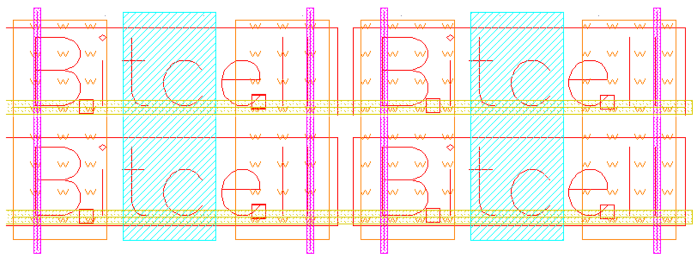
The pictures of the layout of single bitcell and 4 bitcells using the proposed l [1] is shown below with one showing all levels.



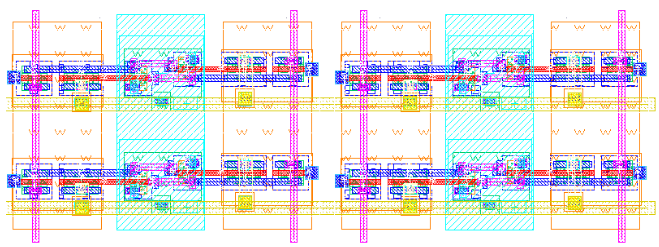
Single bitcell layout [1], WL are connected as such to pass DRC, LVS, in actual 4 bitcells, they are removed.



Single bitcell layout with all levels shown [1]



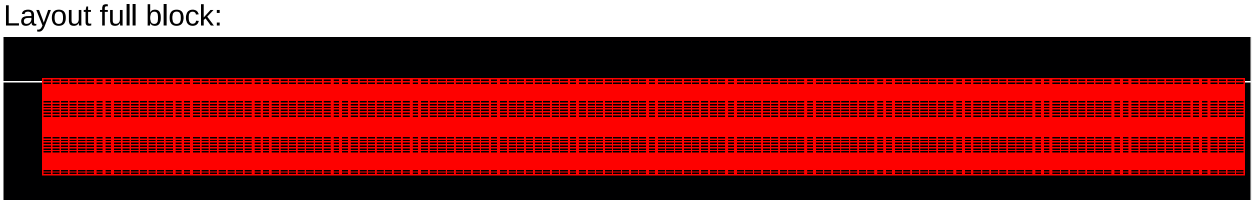
4 bitcells layout



4 bitcells layout the with all levels shown

We have changed the layout of the bitcell from the larger, lower capacitance version proposed earlier to the industry standard 6T design. The reason for doing this is the bitline-capacitance-ignoring effect of the sense amplifier. Since the savings in capacitance no longer have effect on delay or energy used, the only thing that the original design offers is taking up slightly more area. A greater area is a penalty according to the scoring metric we will no longer be pursuing the design proposed in [1].

Layout full block:



Layout (zoom in) partial block:

The layout we will use [2] (same as 2011 Group):

note: the layout shown below was taken from the 2011 groups wiki. Our results should appear similar to, if not identical to theirs since we will be using the same industry standard layout. We don't have our own version quite yet because we decided to change our design yesterday.

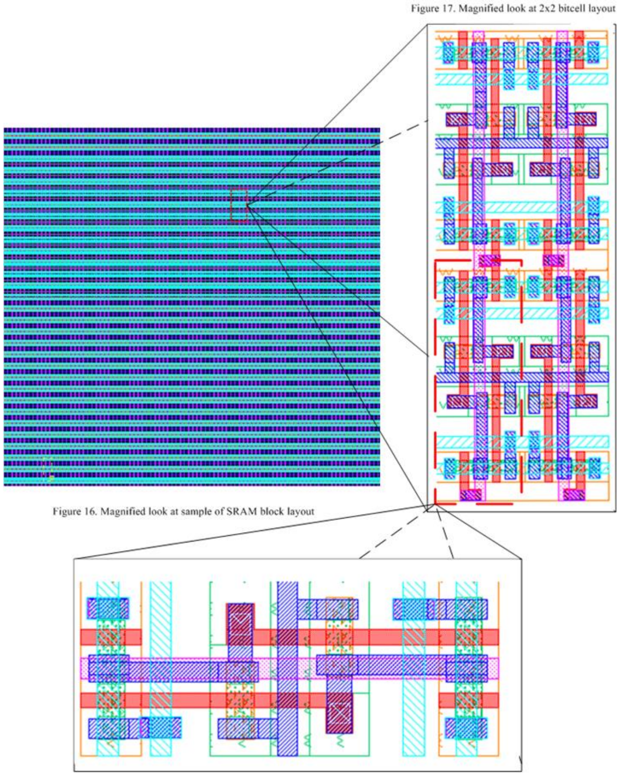


Figure 17. Magnified look at 2x2 bitcell layout

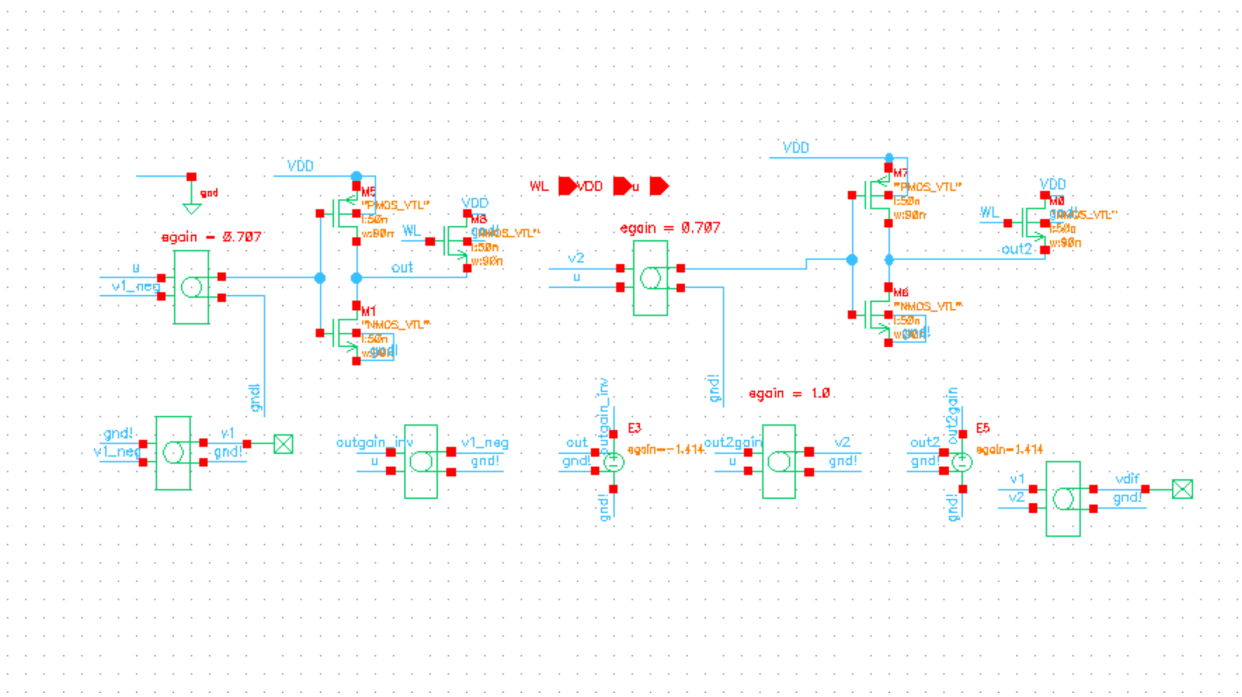
Figure 16. Magnified look at sample of SRAM block layout

Figure 18. Magnified look at bitcell layout

Results

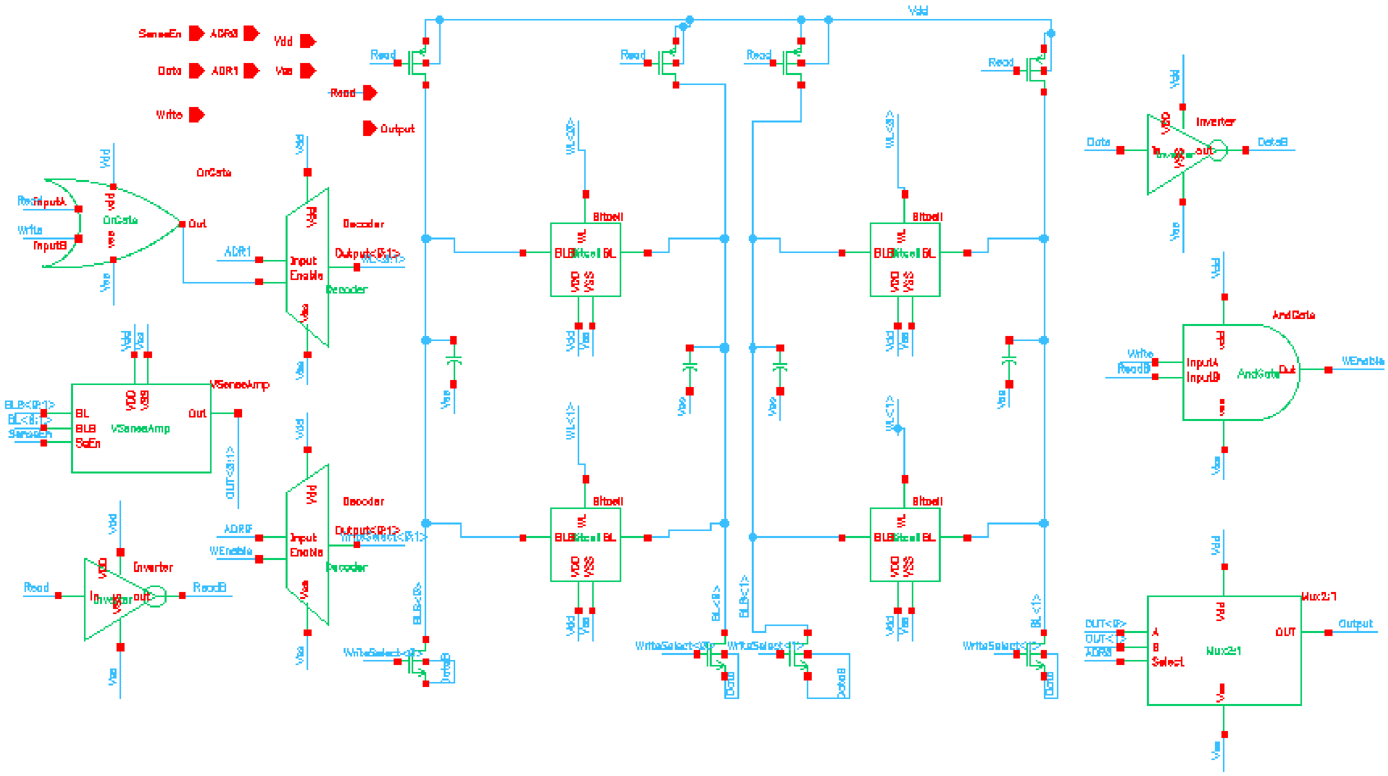
Static Noise Margin:

Schematics were set-up following the tutorial outlined in [3]. Schematics of the circuit is shown below. As VDD is increased, the SNM for all process corners drop off. Each process corner shown below is simulated at 600 mV. Below this voltage noise has the chance of flipping the output of the bitcell. Because of this chance of erroneous output, we chose 600mV as the minimum voltage to supply the bitcell.



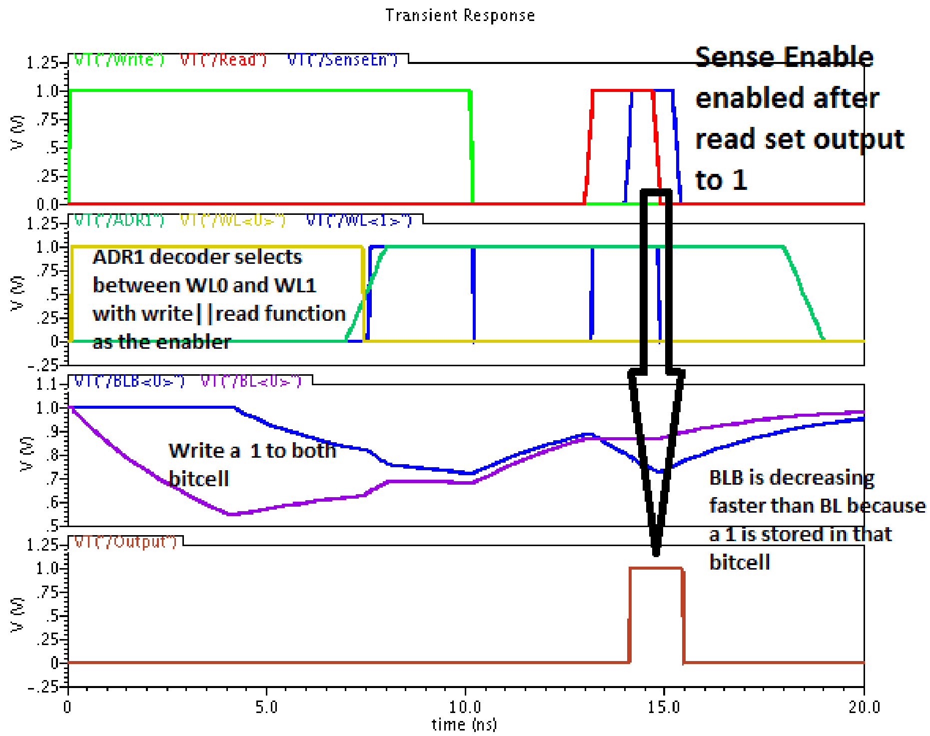
SNM Final values using the netlist (schematics shown above) and ocean scrips in the references

	Write(Max)	Write(Min)	Write SNM	Read(Max)	Read(Min)	Read SNM
FS	334.6	334.62	236.597929	30.04	30.04	14.17042
TT	317.14	317.14	231.322912	143.07	143.15	101.1658
FF	278.18	278.22	196.702964	105.97	105.99	74.93211
SS	333.83	333.75	235.996888	155.55	155.57	110.0046
SF	270.19	270.19	191.053181	209.96	209.96	148.4641



Full Schematics:

Global Corner Simulation (VDD=1.0, T=50C)



References:

[1] Mann, R., Calhoun, B. (2011). *New category of ultra-thin notchless 6T SRAM cell layout topologies for sub-22nm* . 12th International Symposium on Quality Electronic Design

[2] Bailey, S., Linger, K., Lorenzo, R., & Thompson, J. (2011). Team 1 implementation of a low power SRAM design using 45 nm FreePDK technology. Unpublished.

[3] Cabe, A. (2006) ToolsSimulationMemoryStaticNoiseMargin https://venividiwiki.ee.virginia.edu/mediawiki/index.php/ToolsSimulationMemoryStaticNoiseMargin

[4] Chen, Y., Converse, C., Gan, C., & Moore, D. (2010) ClassECE4332Fall10ProjectTeam2 https://venividiwiki.ee.virginia.edu/mediawiki/index.php/ClassECE4332Fall10ProjectTeam2

[5]Wang, J., Lee, H. (1998) “A new current-mode sense amplifier for low- voltage low-power SRAM design”, Eleventh Annual IEEE International Proceeding of ASIC, pp.163-167, Sep. 1998

[6] Blalock, T.N., Jaeger, R.C. (1991) ”A High-speed Clamped Bit-line Current-mode Sense Amplifier”, IEEE J. Solid-State Circuits, vol. 26, no. 4, pp542-548, April 1991

[7] B. S. Amrutur and M. A. Horowitz, “Fast low-power decoders for RAMs,” JSSC, vol. 36, no. 10, 2001.

SNM Netlist

// Generated for: spectre
// Generated on: Nov 12 00:59:20 2013
// Design library name: VLSI_Design_Project
// Design cell name: SNM

```
// Design view name: schematic
simulator lang=spectre
parameters uvoltage=0 pvoltage=0.6 pvoltage2=0.6
E3 (outgain_inv 0 out 0) vcvs gain=-1.414
E5 (out2gain 0 out2 0) vcvs gain=1.414
E8 (vdif 0 v1 v2) vcvs gain=1.0
E7 (v1 0 0 v1_neg) vcvs gain=1.0
E1 (net044 0 v2 u) vcvs gain=0.707 m=1 type=vcvs
E4 (v2 0 out2gain u) vcvs gain=1.0
E6 (v1_neg 0 outgain_inv u) vcvs gain=1.0
E0 (net076 0 u v1_neg) vcvs gain=0.707 type=vcvs
M0 (VDD WL out2 0) NMOS_VTL w=90n l=50n as=9.45e-15 ad=9.45e-15 ps=300n \
        pd=300n ld=105n ls=105n m=1
M8 (VDD WL out 0) NMOS_VTL w=90n l=50n as=9.45e-15 ad=9.45e-15 ps=300n \
        pd=300n ld=105n ls=105n m=1
M6 (out2 net044 0 0) NMOS_VTL w=90n l=50n as=9.45e-15 ad=9.45e-15 ps=300n \
        pd=300n ld=105n ls=105n m=1
M1 (out net076 0 0) NMOS_VTL w=90n l=50n as=9.45e-15 ad=9.45e-15 ps=300n \
        pd=300n ld=105n ls=105n m=1
M7 (out2 net044 VDD VDD) PMOS_VTL w=90n l=50n as=9.45e-15 ad=9.45e-15 \
        ps=300n pd=300n ld=105n ls=105n m=1
M5 (out net076 VDD VDD) PMOS_VTL w=90n l=50n as=9.45e-15 ad=9.45e-15 \
        ps=300n pd=300n ld=105n ls=105n m=1

v0 (VDD 0) vsource dc=pvoltage type=dc
v1 (u 0) vsource dc=uvoltage type=dc
v2 (WL 0) vsource dc=pvoltage2 type=dc
//***!!!!!! ABOVE LINE Commented = Hold unCommented=Read
```

SNM Ocean Script

```
;; We want to print some text outputs containing measured data
;; DEFINE OUTPUT PRINT FILE ('w' means open for writing)
of = outfile( "output.txt" "w" )

;; SET SIMULATOR
simulator( 'spectre )

;; SET DESIGN
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; NOTE: the design name can only be 'netlist'.
;; We have had trouble using other names.
;; However, the directory name can be added, like
;;  "./mydir/netlist"
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
design( "netlist" )

;; Now we need to define the models for the transistors
;; Here we hardcode the path to the model library
;; DEFINE THE MODEL
libdir = "./freepdk.scs" ; LIBRARY PATH

;; Set parameters specific to this model library to select the right model
deviceOption = "lvt"
CornerList = list("SS"  "TT" "FS" "FF" "SF" ); = THE GLOBAL CORNER CASES

;; Define simulation options - these may differ depending on your requirements
option( 'gmin 1e-18 'iabstol 1e-12)


;; Define simulation options - these may differ depending on your requirements
option( 'gmin 1e-18 'iabstol 1e-12)

;; SET VDD - here we define the parameter pvdd that we used in netlist


;; SET MODEL FILE
;foreach( Corner CornerList
;;desVar( "pvoltage" 0.6)
;;desVar( "pvoltage2" 0.6)
Corner = "SF";
pvoltage=0.6
pvoltage2=0.6
model = strcat( deviceOption Corner )
modelFile( list(libdir "normStat") list(libdir model) )

temp( 25 )

sprintf(dir "./%s" model)
resultsDir( dir )
;; SPECIFY ANALYSIS
 analysis('dc ?param "uvoltage" ?start -pvoltage ?stop pvoltage ?step 0.001  ?errpreset
'conservative)

; NB - Save only specific signals to save time and space for very large simulations

; RUN THE SIMULATION
;;monteCarlo( ?numIters 10 ?analysisVariation 'processAndMismatch ?sweptParam
"None" ?sweptParamVals "25" ?saveData t )
```

```
;; MONTE CARLO EXPRESSION FOR MEASUREMENTS (Remember to put the monteexpr's
BEFORE the monterun.
;; Or else the expression values would not get printed in the mcdata file)
;;monteExpr( "rt" "riseTime(v(\"out\" ?result 'tran) 0.5n t 1.5n t 10 90)" )
;;monteExpr( "ft" "riseTime(v(\"out\" ?result 'tran) 1.5n t 2.5n t 10 90)" )

;; START MONTE CARLO SIMULATION


save( 'all )
;;monteRun()
run()
;); end CornerList
selectResults('dc)
plot(v("v1") v("v2") v("vdif"))
```

SNM Curves

Figures 6-10 are showing the butterfly curves generated by our simulations in ocean to test the Hold SNM values for 5 different process corners. The blue and red plots stand for V1 and V2. They show butterfly curves transposed onto the “u-v” axis. The lower pink curve is showing the difference between the upper two curves. By take the absolute value of the max and min points of the pink curve, then multiply the smaller of the two by 1/sqrt(2) to obtain the SNM value of the cell. (multiply by this factor because we want the length of the side of the square, not the diagonal). Figures 11-15 are showing those curves for read.

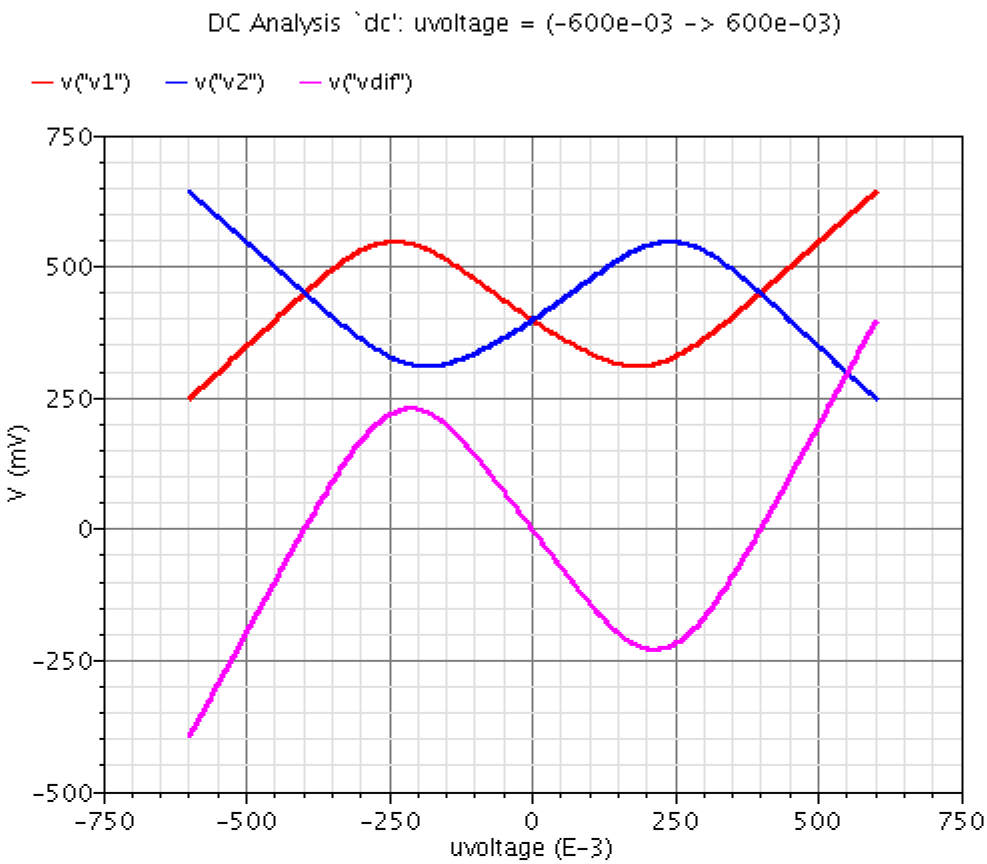


Figure 6. This is a graph for Hold SNM at FF corner.

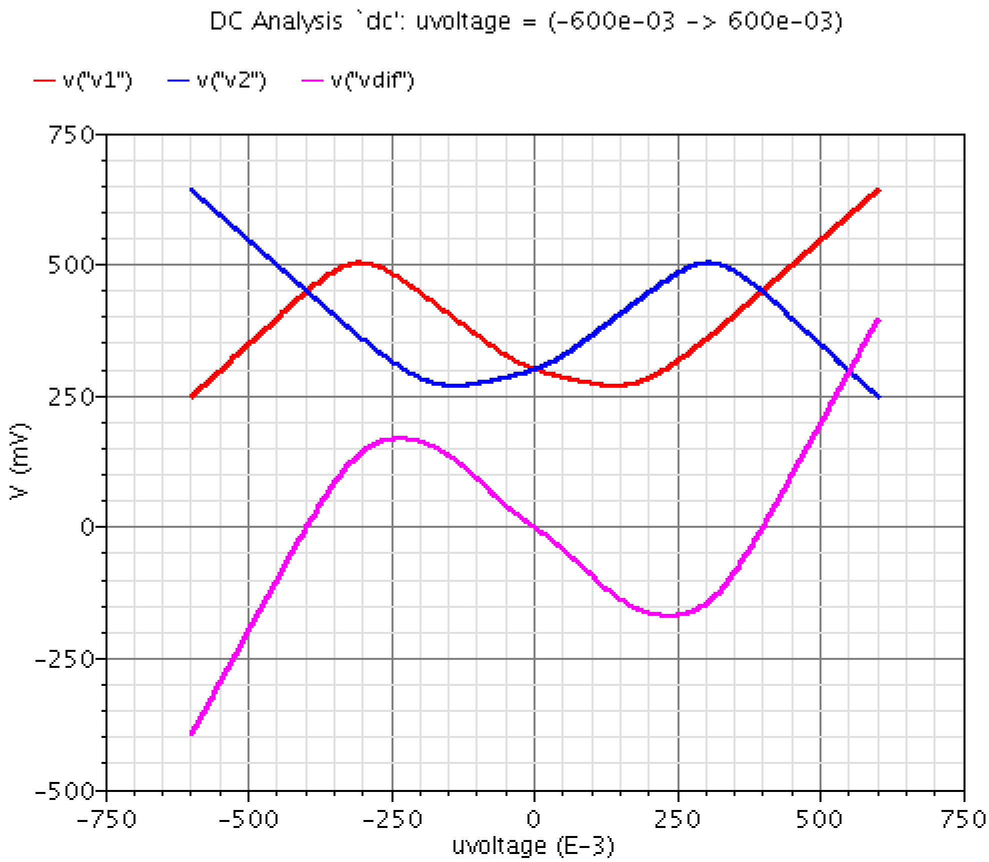


Figure 7. This is a graph for Hold SNM at FS corner.

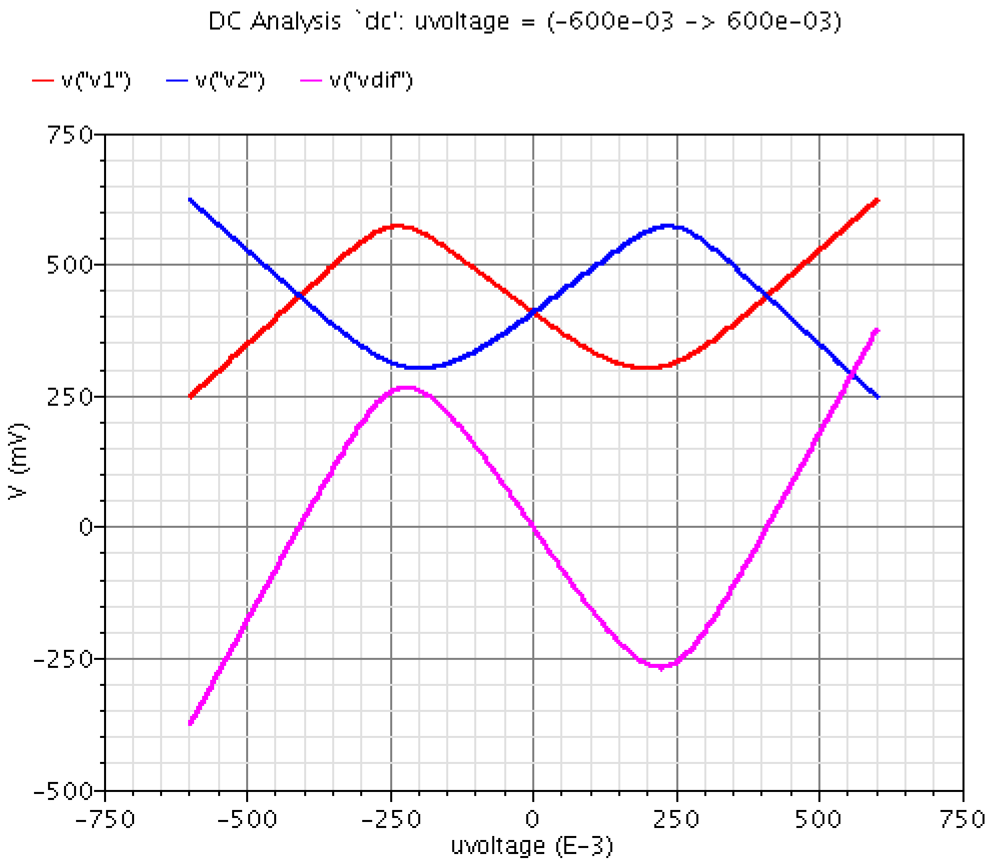


Figure 8. This is a graph for Hold SNM at TT corner.

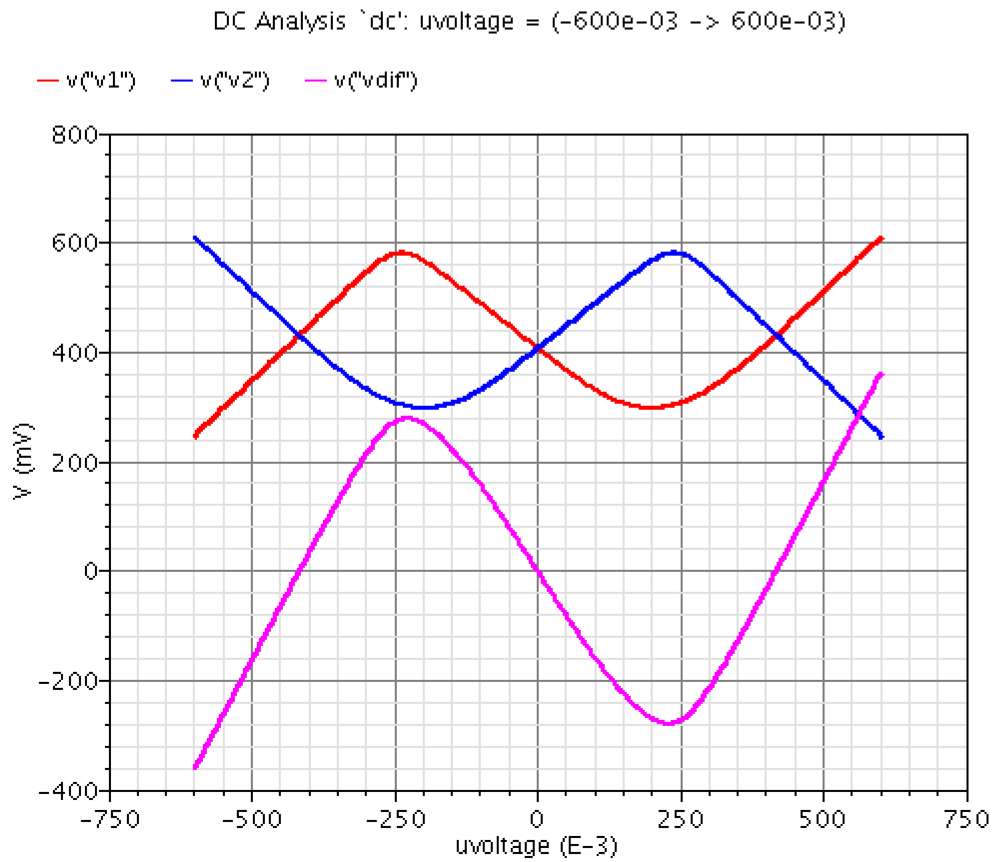


Figure 9. This is a graph for Hold SNM at SS corner.

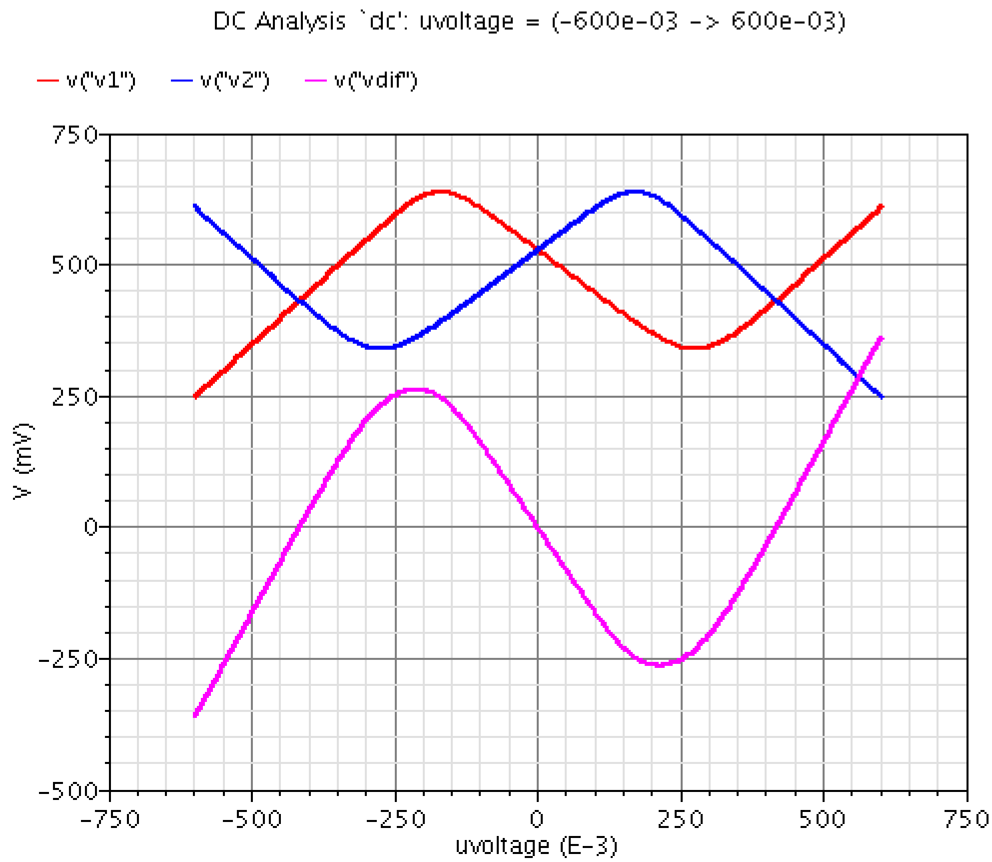


Figure 10. This is a graph for Hold SNM at SF corner.

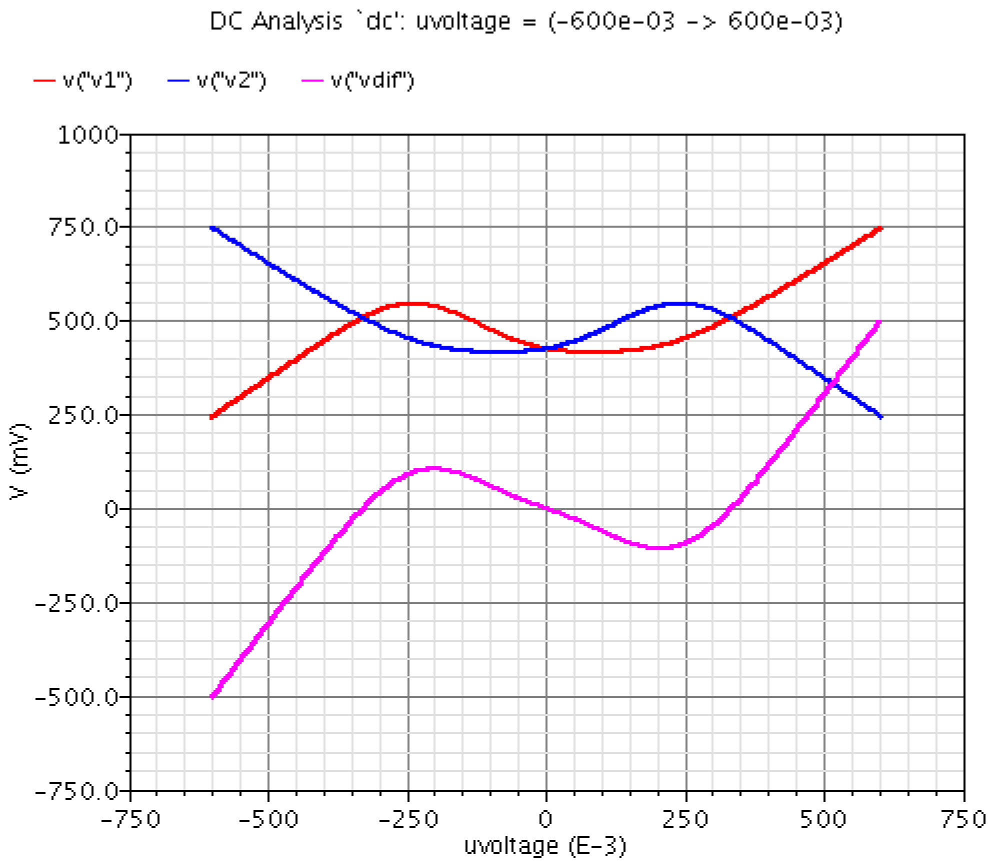


Figure 11. This is a graph for Read SNM at FF corner.

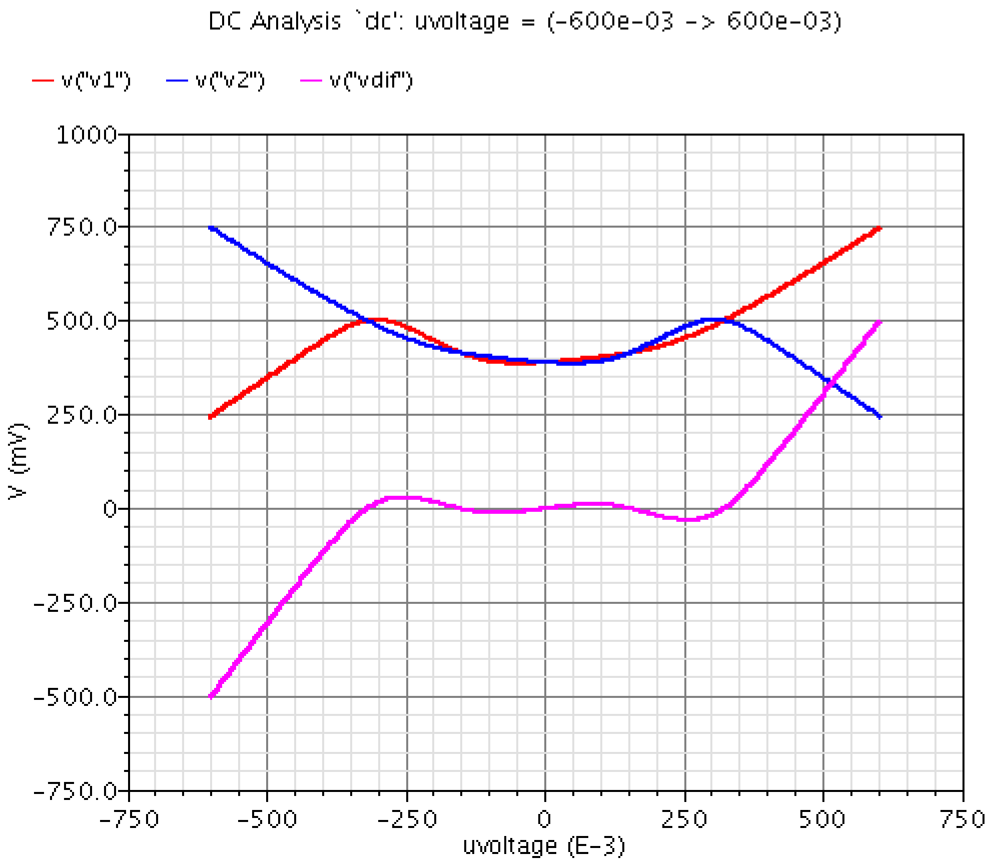


Figure 12. This is a graph for Read SNM at FS corner.

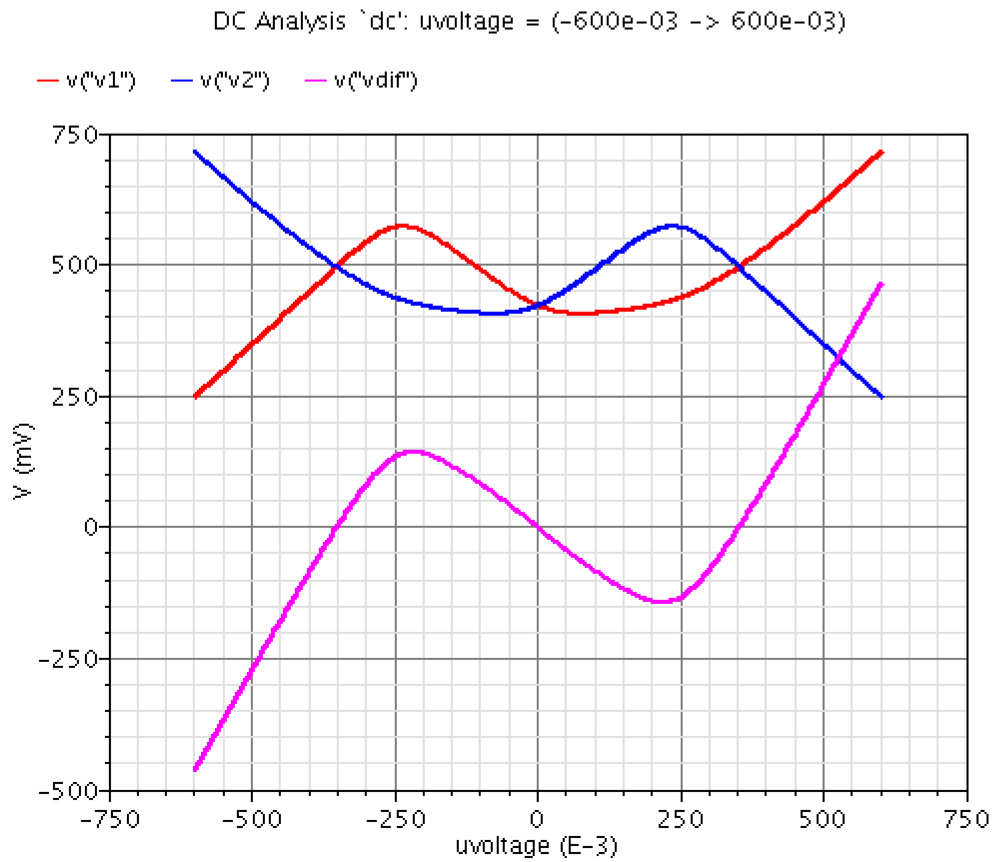


Figure 13. This is a graph for Read SNM at TT corner.

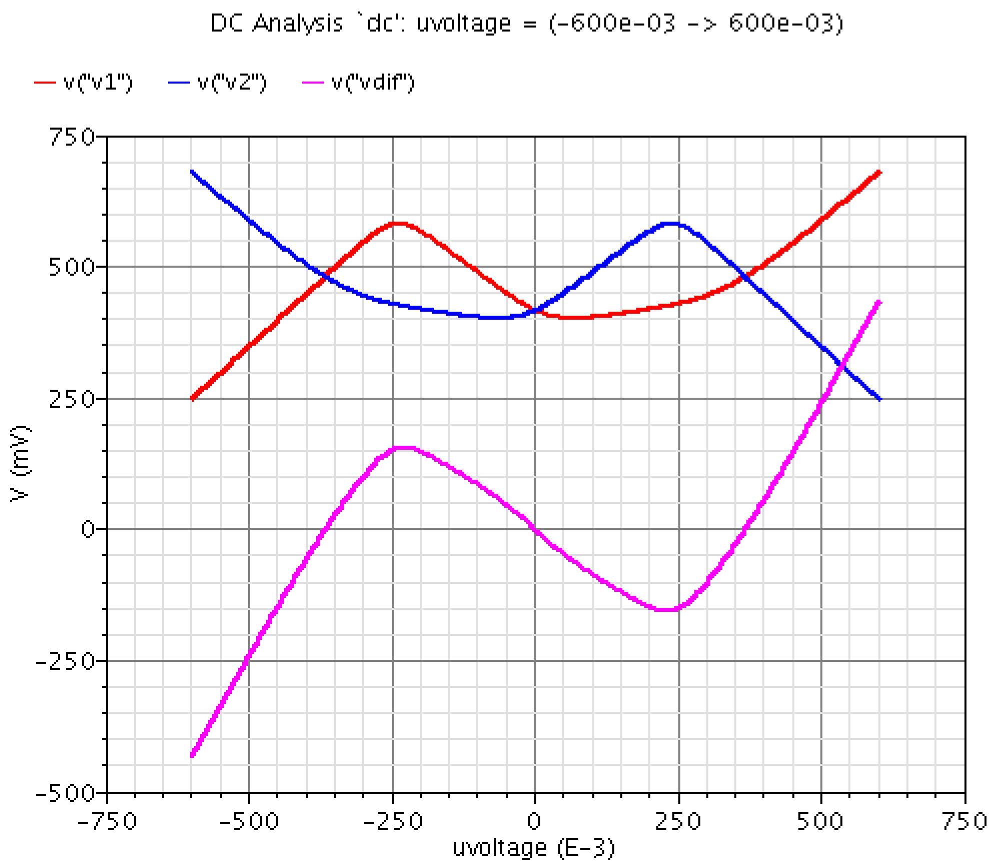


Figure 14. This is a graph for Read SNM at SS corner.

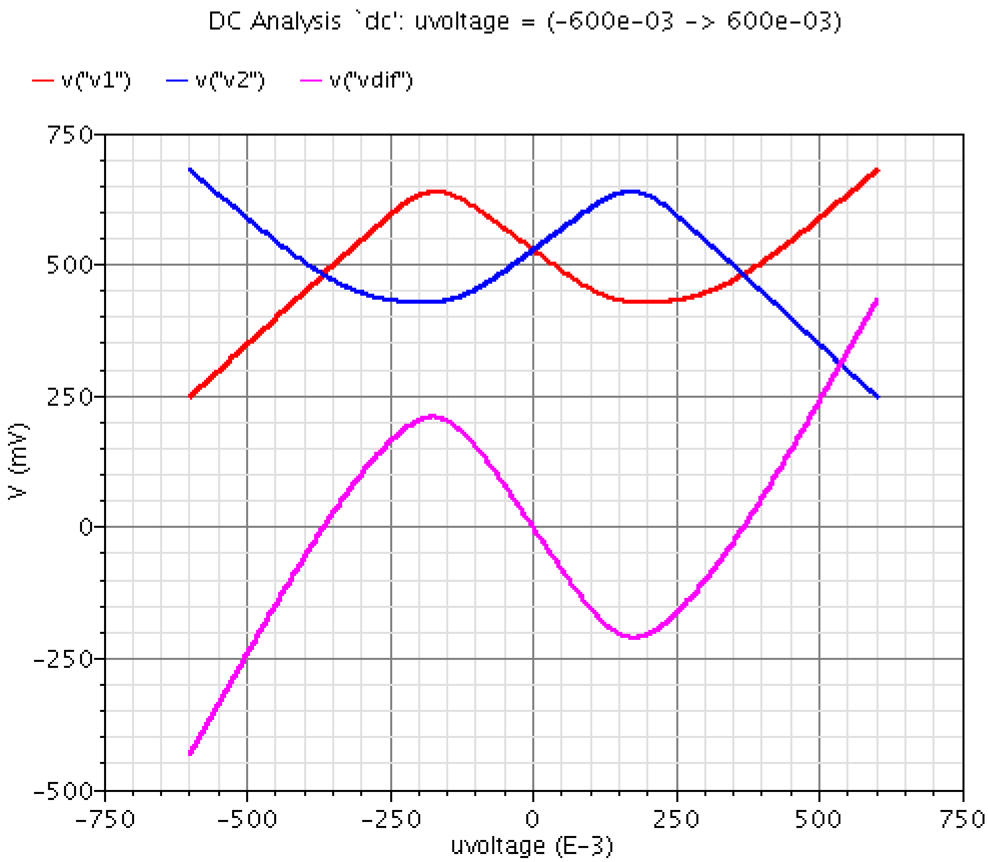


Figure 15. This is a graph for Read SNM at SF corner.